

游荡在思考的迷宫中

King-Yin Yan (甄景贤), Ben Goertzel, and Juan Carlos Kuri Pinto

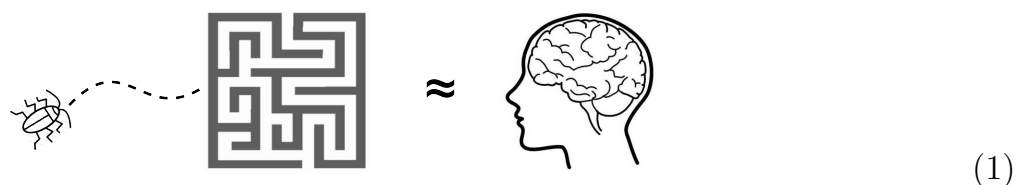
General.Intelligence@Gmail.com

Abstract. 介绍一个基於 增强学习和 深度学习的极简约的 cognitive architecture，它在数学上是一个 Hamiltonian 系统，而其 Lagrangian 对应於智能系统的「奖励」或「欲望」的价值。经典逻辑 AI 的技巧可以搬到这个 setting 之下，而连续时间化之后，可以用上微分几何的技巧。传统的「逻辑 AI 知识表述」被新的表述法取代，后者的结构是由神经网络的深度学习「诱导」出来的。

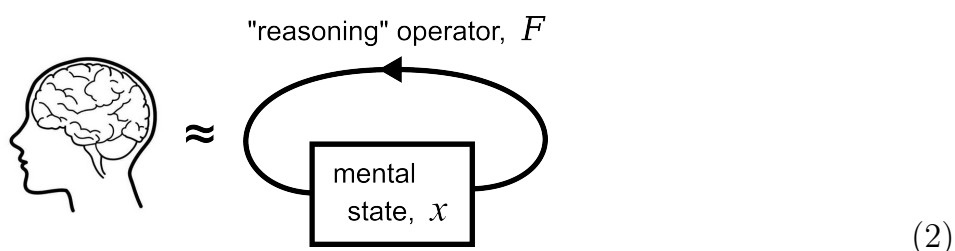
本文介绍一些（非原创的）已知理论，然后指出一些新的研究方向。

0 中心思想

标题中的比喻是指用增强学习的方法控制一隻自主的智能系统 (autonomous agent)，在「思维空间」中寻找最优路径：



关键是将「思考」看成是一个动态系统 (dynamical system)，它运行在思维状态 (mental states) 的空间中：



举例来说，一个思维状态可以是以下的一束命题：

- 我在我的房间内，正在写一篇 AGI-17 的论文。
- 我正在写一句句子的开头：「举例来说，」
- 我将会写一个 NP (noun phrase)：「一个思维状态....」

思考的过程就是从一个思维状态 **过渡** (transition) 到另一个思维状态。就算我现在说话，我的脑子也是靠思维状态记住我说话说到句子结构的哪部分，所以我才能组织句子的语法。

以下三者其实是同义词：

- 在人工智能里叫 **强化学习** (reinforcement learning (RL))
- 在运筹学里叫 **动态规划** (dynamic programming)
- 在现代控制论 (control theory) 中的 **状态空间** 表述

1 控制论：动态系统

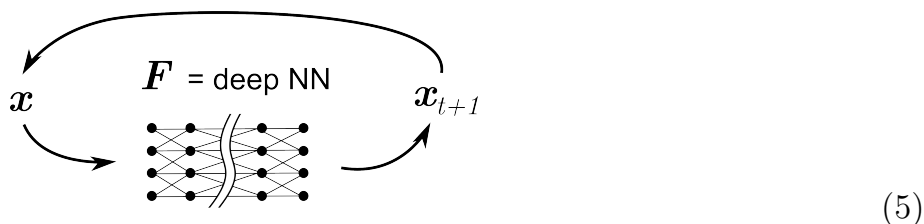
思维状态是一支向量 $\mathbf{x} \in \mathbb{X}$ ， \mathbb{X} 是所有可能的思维状态，思考算子 (reasoning operator) \mathbf{F} 是一个 endomorphism 映射： $\mathbb{X} \rightarrow \mathbb{X}$ 。

在数学上这是一个标准的**动态系统 (dynamical system)**，它可以用以下方法定义：

离散时间	$\mathbf{x}_{t+1} = \mathbf{F}(\mathbf{x}_t)$	(3)
------	---	-----

连续时间	$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$	(4)
------	---	-----

在我设计的 cognitive architecture 里， \mathbf{F} 用深度神经网络来表示（所谓「深度」无非是很多层的意思）：



一个**神经网络**是一个有很多参数的非线性算子：

神经网络 $F(\mathbf{x})$	$= \bigcirc(W_1 \bigcirc(W_2 \dots \bigcirc(W_L \mathbf{x})))$	(6)
----------------------	--	-----

其中 W_ℓ 是每一层的**权重矩阵**， \bigcirc 是一个 sigmoid 形状的非线性函数。

如果连续时间的话 \mathbf{f} 也可以用深度神经网络表示，不过这两个 \mathbf{F} 和 \mathbf{f} 的性质是不同的，它们之间由这个关系决定： $\mathbf{x}(t+1) = \mathbf{F}(\mathbf{x}(t))$ 。为方便起见，我会随意使用连续或离散时间的表述。

控制系统 (control system) 和动态系统的分别是在定义中加入了 **控制向量 $\mathbf{u}(t)$** ：

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \quad (7)$$

控制论的目的就是找出最好的 $\mathbf{u}(t)$ 函数，令系统由初始状态 \mathbf{x}_0 去到终点状态 \mathbf{x}_\perp 。

一个典型的控制论问题是这样描述的：

状态方程	$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), t]$	(8)
------	---	-----

边值条件	$\mathbf{x}(t_0) = \mathbf{x}_0, \mathbf{x}(t_\perp) = \mathbf{x}_\perp$	(9)
------	--	-----

目标函数	$J = \int_{t_0}^{t_\perp} L[\mathbf{x}(t), \mathbf{u}(t), t] dt$	(10)
------	--	------


要找的是最优控制 $\mathbf{u}^*(t)$ 。

根据控制论，最优路径的条件，是由 Hamilton-Jacobi 方程给出：

$$\boxed{\text{Hamilton-Jacobi equation}} \quad 0 = \frac{\partial J^*}{\partial t} + \min_u H \quad (11)$$

跳过一节之后我会解释 J ， L ，和 H 的意义。

2 强化学习 / 动态规划

Reinforcement learning 是机器学习里面的一个分支，特别善于控制一只能够在某个环境下 **自主行动** 的个体 (autonomous agent)，透过和 **环境** 之间的互动，例如 sensory perception 和 rewards，而不断改进它的 **行为**。强化学习最典型的比喻是一只迷宫中寻找食物和避开敌人的小昆虫：

一个强化学习的系统由 4 个元素的 tuple 构成

$$\boxed{\text{强化学习系统}} = (\text{States} \ni \mathbf{x}, \text{Actions} \ni \mathbf{u}, R = \text{Rewards}, \pi = \text{Policy}) \quad (12)$$

详细可参阅我写的《强化学习 tutorial》。

U 是一连串行动的 rewards 的总和：

$$\boxed{\text{状态 } 0 \text{ 的总价值 } U(\mathbf{x}_0)} = \sum_t \boxed{\text{时间 } t \text{ 时的奖励 } R(\mathbf{x}_t, \mathbf{u}_t)} \quad (13)$$

例如说，行一步棋的效用，不单是那步棋当前的利益，还包括走那步棋之后带来的后果。例如，当下贪吃一只卒，但 10 步后可能被将死。又或者，眼前有美味的食物，但有些人选择不吃，因为怕吃了会变肥。

Dynamic programming 的中心思想是 **Bellman optimality condition**。Richard Bellman 在 1953 年提出这个方程，当时他在 RAND 公司工作，处理的是运筹学的问题。

Bellman equation 说的是：「如果从最佳选择的路径的末端截除一小部分，余下的路径仍然是最佳路径。」

$$\boxed{\text{全路径的价值}} = \max_u \{ \boxed{\text{在当前状态下选择 } \mathbf{u} \text{ 的奖励}} + \boxed{\text{余下路径的价值}} \} \quad (14)$$

$$\boxed{\text{Bellman 方程}} \quad U^*(\mathbf{x}) = \max_u \{ R(\mathbf{u}) + U^*(\mathbf{x}_{t+1}) \} \quad (15)$$

这条看似简单的式子是动态规划的**全部内容**。它的意义是：我们想获得最佳效益的路径，所以将路径切短一些，於是问题化解成一个较小的问题；换句话说它是一个 **recursive relation**。

在人工智能中常用一个 trick，叫 **Q-learning**。 Q 值是 U 值的一个变种； U 是对每个 state 而言的， Q 把 U 值分拆成每个 state 中的每个 action 的份量。换句话说， Q 就是在状态 \mathbf{x} 做动作 \mathbf{u} 的 utility。 Q 和 U 之间的关系是：

$$U(\mathbf{x}) = \max_u Q(\mathbf{x}, \mathbf{u}) \quad (16)$$

Q 的好处是方便学习，只需要学习在每一个状态下选择哪个动作的价值；亦即所谓“model free”学习。

在强化学习的框架下，智能系统的运作可以分开成两方面：**思考** 和 **学习**。

- **思考**即是根据已学得的知识（知识储存在 deep NN 里），在思维空间中找寻 \mathbf{x} 最优的轨迹，方法是根据 Bellman 方程计算 \mathbf{u}^* 。 \mathbf{x} 的轨迹受 deep NN 约束（亦即是说，系统只能依据正确的知识去思考），思考时 deep NN 是不变的。
- **学习**就是学习神经网络 deep NN 的 weights W_ℓ ，改变 W 即改变 \mathbf{F} ，而 \mathbf{F} 决定状态方程 (3)，所以整个系统变了另一个系统。换句话说，deep NN 的学习是一种 **second-order learning**：考虑两个系统 \mathbf{F} 和 $\mathbf{F} + \epsilon \mathbf{F}_0$ ，经过很多次思考过程，如果奖励的平均值在后者有所增加，则 \mathbf{F} 向 \mathbf{F}_0 方向学习。

3 控制论与强化学习的关系

在**强化学习**中，我们关注两个数量：

- $R(\mathbf{x}, \mathbf{u})$ = 在状态 \mathbf{x} 做动作 \mathbf{u} 所获得的 **奖励** (reward)
- $U(\mathbf{x})$ = 状态 \mathbf{x} 的 **效用** (utility) 或 **价值** (value)

简单来说，「价值」就是每个瞬时「奖励」对时间的积分：

$$\boxed{\text{价值 } U} = \int \boxed{\text{奖励 } R} dt \quad (17)$$

用**控制论**的术语，通常定义 cost functional：

$$\boxed{\text{价钱 } J} = \int L dt + \Phi(\mathbf{x}_\perp) \quad (18)$$

其中 L 是“running cost”，即行走每一步的「价钱」； Φ 是 terminal cost，即到达终点 \mathbf{x}_\perp 时，那位置的价值。

在**分析力学**里 L 又叫 **Lagrangian**，而 L 对时间的积分叫「作用量」：

$$\boxed{\text{作用量 (Action) } S} = \int L dt \quad (19)$$

Hamilton 的**最小作用量原理** (principle of least action) 说，在自然界的运动轨迹里， S 的值总是取稳定值 (stationary value)，即比起邻近的轨迹它的 S 值最小。

Hamiltonian 的定义是 $H = L + \frac{\partial J^*}{\partial \mathbf{x}} \mathbf{f}$ ，它是由 Lagrange multiplier 的方法走出来的。详细可参看我写的《控制论 tutorial》。

其实它们讲的是同一样东西，所以有如下的对应：

强化学习	最优控制	分析力学
效用/价值 U	价钱 J	作用量 S
即时奖励 R	running cost	Lagrangian L
action a	control u	(外力?)

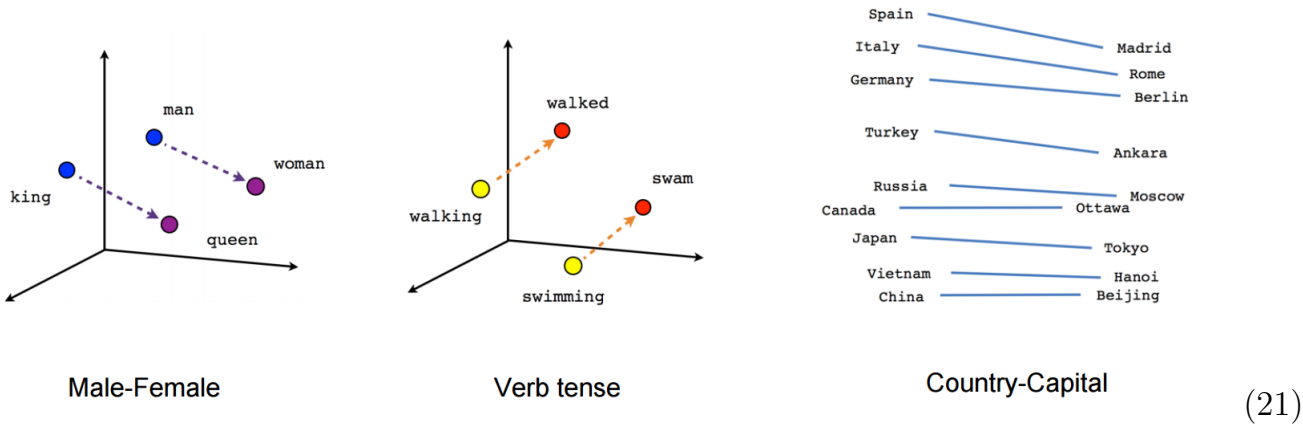
(20)

有趣的是，奖励 R 对应於力学上的 Lagrangian，其物理学单位是「能量」；换句话说，「快感」或「开心」似乎可以用「能量」的单位来量度，这和通俗心理学里常说的「正能量」不谋而合。而，长远的价值，是以 [能量 \times 时间] 的单位来量度。

4 经典逻辑 AI

在系统的状态方程 (3) 中， F 是可以自由变动的（ F 代表被学习的知识），换句话说，整个系统几乎没有结构。在无限维的泛函空间搜寻 F 是不切实际的，所以要引入逻辑 AI 的结构，令 F 的搜寻范围缩小。在机器学习中这种做法叫 inductive bias，是加快学习的必经之路。这个问题会在我的论文《神经与逻辑之间的桥》探讨（先前发表了初稿，但那里有不少漏洞）。

换句话说：我们的目的是想将整套逻辑 AI 的器材搬到 连续 的微分流形上去实现。这个做法，是受了 Google 的 Word2Vec [1] 算法的启发，它可以将自然语言的词语 embed 到度量空间中，而且词语之间的距离是 semantic distance（语义学上的距离）：



当然，word2vec 一经发表之后，很多人开始构思怎样把「句子」也 embed 到度量空间上。第一个想法当然是用 **tensor product**，例如 “he loves her” 这句话就变成了 $he \otimes loves \otimes her$ 。但这个做法很有问题；例如『人不可貌相』和『金玉其外 败絮其中』这两句句子意义相近，但表面上 (syntax) 却很不同。用 tensor product 的做法，得出的空间结构是 syntactic 而不是 semantic 的，但神经网络学习的重点是 泛化 (generalization)，它基於「邻近的点的意义相近」才能成功，所以一定要 semantic distance。

有一个新的想法是：不要看状态空间 \mathbb{X} 的结构，因为这牵涉到逻辑原子概念怎样组合成逻辑句子的 composition 问题，这 composition 如果是 algebraic 的就会产生上面的 syntactic 问题。解决办法是：考虑 $\mathbb{X} \rightarrow \mathbb{X}$ 之间的函数。这些函数的 环 反过来决定 \mathbb{X} 的结构。Nestruev 2003 的书《Smooth Manifolds and Observables》有讲述类似的观点。

换句话说， $F : \mathbb{X} \rightarrow \mathbb{X}$ 应该由两个家族合并而成：一个是根据 Bellman 的学习算子，另一个是逻辑的学习算子。

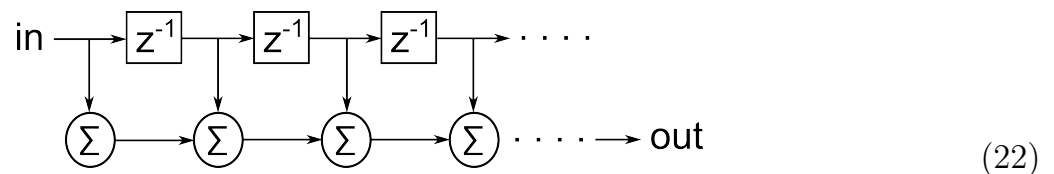
5 Episodic memory

设计了这个 minimalist architecture 之后，发现比起人脑有个严重缺陷：没有 **事件记忆** (episodic memory)。换句话说，只能留意当下发生的事件，但不能记住一段**故事**。

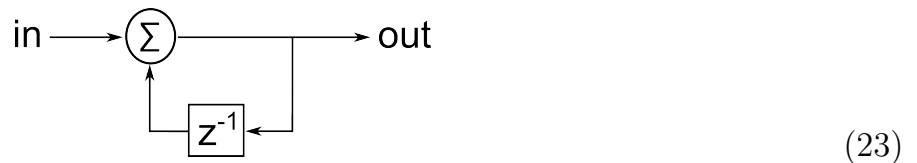
这牵涉到「什么是记忆？」的问题。在 minimal architecture 里， F 代表“static knowledge”，亦即（相对地）**永恒不变**的知识 / 规律，而 x 代表当下的**状态** / 短期记忆，亦即“dynamic knowledge”。Episodic memory 介乎「长期」与「短期」记忆之间。

深度学习在近年很厉害，但 deep NN 的缺点是没有**记忆**。有限自动机 (finite state machines) 不是**全能**的计算器，它和 Turing machine 的分别就是缺少了那条「记忆磁带」。所以，深度神经网络 + 记忆 就可以变成 universal 的计算器。如何设计「可微分」的记忆是一重要课题，因为可微分的记忆可以用 gradient descent 学习。

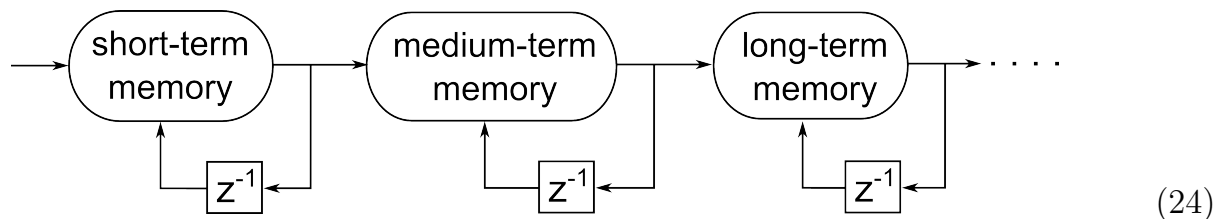
在 **信息处理** (signal processing) 理论中， z -transform 以 z^{-1} 可以代表 **时间延迟 1 步** (1-step time delay unit)。用一连串的 z^{-1} 可以组成长度为 k 单位的记忆：



以上的无穷串列可以用这个 recursive 结构代替：



多层的 hierarchical 记忆或许可以这样实现：



Z -transform 的**连续时间**版本就是 Laplace transform。

信号处理的资深研究者 Simon Haykin 最近也用 RL + 记忆 的设计，详见他的 2012 新书《Cognitive dynamic systems》。

References

1. Mikolov, Sutskever, Chen, Corrado, and Dean. Efficient estimation of word representations in vector space. *Proceedings of workshop at ICLR*, 2013.
2. Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.